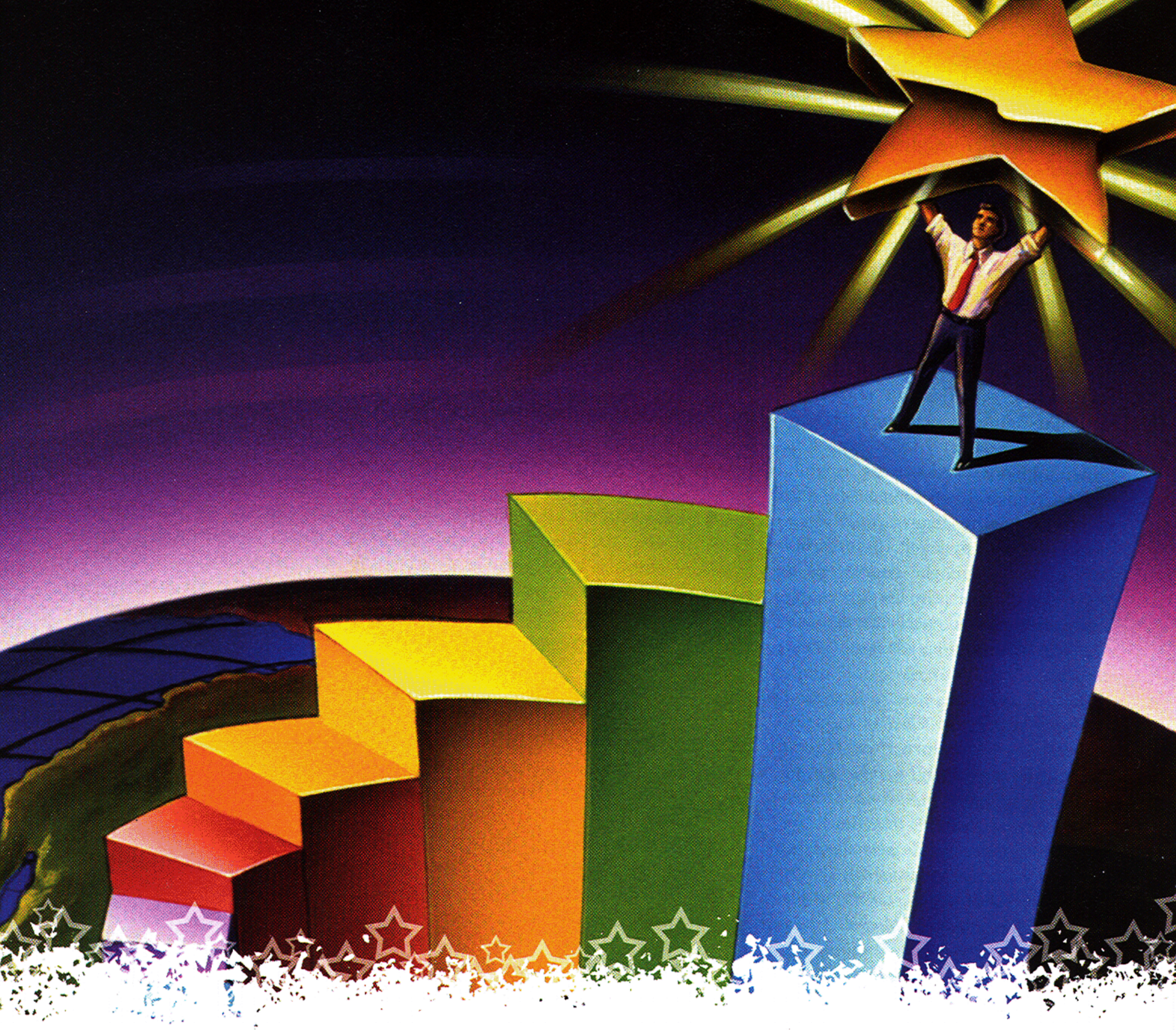

PROVING OUR WORTH

Quantifying the Value of Testing

by Lee Copeland

"Testers are such a nuisance. They waste time, create bugs, get in the way of product shipment, lower the morale of our developers, and cost too much." That's how my interview with Rajan began. Rajan is a project manager for MegaCorp, my new consulting client. I'd been invited by MegaCorp's executive management to perform a test process assessment to assist in improving their overall testing process. Later in the day, as I was talking with Rayanne, the testing manager, she countered, "Management doesn't have a clue. They don't see the value in our work. And they can't make up their minds—one day quality is the most important thing, the next day shipping the product is."

I hear these verbal bombardments a lot. Perhaps you have heard them too. Perhaps they are flying overhead in your organization right now. How do these critical feelings arise? I believe this antagonism often comes from a basic misunderstanding of the real purpose of testing.

In his seminal book *The Art of Software Testing*, Glenford Myers defined testing as "the process of executing a program or system with the intent of finding errors." Bill Hetzel expanded the definition: "Testing is any activity aimed at evaluating an attribute of a process or a system." More recently, Rick Craig and Stefan Jaskiel wrote, "Testing is a concurrent lifecycle process of engineering, using, and maintaining testware in order to measure and improve the quality of the software being tested." Note the similarities—the processes of finding, evaluating, measuring, and improving. For a quarter of a century, we as testers have focused on the internal process of testing, while generally disregarding its real purpose.

The Real Purpose of Testing

The real purpose of testing is to create information. James Bach nailed it when he wrote, "The ultimate reason testers exist is to provide information that others on the project use to create things of value." That is why testing exists—to provide information of value. So, when managers complain that testing "costs too much," perhaps they are really trying to say, "I'm not getting enough valuable information to justify the cost of testing." When testers say, "My management doesn't see the value in our work," perhaps they are really trying to say, "My management doesn't value the information I'm providing to them." To prove our worth and to increase the value of testing, we must first focus on testing's purpose—providing valuable information—not its process.

The Value of Testing

The value of testing is the value of the information we create for others. We do not determine its value—our clients do. Author and CEO Kamran Parsaye wrote that in a utopian world, the value of information would be "measured by observing differences in the decision makers' performances when provided with different types of information." But we do not live in a world where we provide information to one group and withhold it from another in a kind of "value of information" experiment. In our world as testers, we provide information to clients who will—we hope—be receptive. But not all will be. To them, if the information has no value, then testing has no value. For those open to the information we provide, determining its value is still not an easy task.

Some argue that testing has a value independent of the information that it creates. Every time testing discovers a defect that is repaired before it escapes to the customer, we have saved money for our organization and therefore created value. While that is true, most testing organizations do not quantify that value and thus cannot communicate it to executive management. Rather than having solid economic data, all we have are "campfire stories," anecdotes that may be interesting around the campfire but that are ineffective in the boardroom.

GQM to the Rescue

What information would our clients find valuable? After identifying potential clients we usually begin inundating them with long lists of data items. Instead, let's consider a different approach—the Goal Question Metric (GQM) model originated by Dr. Victor Basili. First, we choose a specific client that might find our information useful. Then, using the GQM model, we ask members of the group, "What are your quality-related *goals*?" What are you striving for? What are you focused on? What are you rewarded for? What are you *really* rewarded for?" The next step is to ask, "What *questions* must you be able to answer in order to know whether you are meeting your quality-related goals or not?" The final step in using the GQM model is to ask, "What information (*metrics*) would you need in order to answer the questions about your goals?" Thus, rather than starting with a large pile of possibly useful information, we supply the information most needed and valued by our client.

You could be lucky when you ask the GQM questions—your clients have already thought about these questions and have answers for you. Or you could be unlucky and watch them struggle to articulate the Gs and Qs. One trick I've used to get clients "un-stuck" is to ask them to consider these questions:

- What information would you like to know about the losses, financial and otherwise, that could affect our organization because of the quality aspects of our products?
- What information would you like to know about the likelihood of these losses? Could we mitigate them through control or prevention?
- How much would you be willing to pay for such information? If the cost of creating the information is less than the expected loss, gathering that information may be a good financial business decision.
- What are the possible remedial actions open to our organization? Can we take positive actions to reduce or minimize these losses or must we just ride out the storm?

The Magic Information Machine

Another trick I've used is to tell them about the Magic Information Machine. Pose this question to your clients, "If you had a machine that would answer your quality questions regarding the systems you're developing, what information would you like it to tell you?" Using the clients' answers to this question, you could design a testing process to deliver that information. If they don't figure it out, tell them that the Magic Information Machine is testing. That is its purpose—to provide valuable information.

The Value of Information

The value of information is derived from its usefulness in making informed decisions. Aside from specific types of information, what would be general attributes of all—or almost all—valuable information?

Accurate

Lookout on the Titanic: "There are no icebergs in the vicinity, Captain."

The information we provide must be correct, exact, and factual. In a word, it must be trustworthy. Critical decisions may be based on it. Sometimes we cannot supply exact information, but we can guarantee its accuracy within a range. If that is the case, make sure the range is communicated as part of the information. Less than exact information may be better than no information at all.

Timely

Lookout on the Titanic (as the ship is sinking): "There are icebergs in the vicinity, Captain."

The information we supply must be available at a time when its application can affect decisions. Accurate information, arriving too late, can be valueless.

Complete

Lookout on the Titanic: "There are no icebergs on the port side, Captain."

Information must be complete; it must tell the whole story. If we cannot guarantee its completeness, we must give an indication of its scope of applicability.

Relevant

Lookout on the Titanic: "Look, Captain, the northern lights. Aren't they beautiful tonight?"

The world is full of interesting, accurate, timely, complete information that is not relevant to the decisions at hand—don't include it in the information you supply to your clients.

Unique

Lookout on the Titanic: "It's cold out here tonight, Captain."

Information that is already known or that is readily available from other sources is generally not valued. It's just redundant. However, receiving the same information from multiple sources may be an indicator of its importance.

Actionable

Lookout on the Titanic: "Captain! Icebergs on the starboard side, coming at us fast!"

Important decisions can be made based on this information. Our clients are choosing from a number of actions, and this information allows them to make more effective choices.

Client-oriented

Finally, the information must be expressed in terms that are meaningful and useful to our clients including loss of sales, revenue, market share, and reputation. This also means claims for compensation, restorative actions, repair, and redistribution are important. It is not useful to report the number of test cases written, test cases executed, Severity 2 defects found, or the myriad of internal process metrics we often dump on them.

Proving Our Worth—Quantifying Our Value

We may not perform the Parsaye "value of information" experiment, but we can still quantify the value of the information that we, as testers, provide. Let's listen in on the private thoughts of a developer, test manager, and project manager:

DEVELOPER

GOAL: My goal is to get my code out on time without making any real blunders in it; I don't want the customer finding big defects. I'll look foolish in front of the whole team, especially my manager.

QUESTIONS:

- How many defects have I created?
- Are my defects scattered randomly throughout the code or are they clumped in a few modules?
- Am I making random mistakes or am I creating the same type of defects again and again?

METRICS:

- Number of defects
- Defect distribution
- Defect root cause

VALUE: Quantifying the value of this information is difficult for me. Having information about the defects I create could help me make fewer mistakes in the future. If it costs "X" dollars to find and fix each one of my defects, and if I made "Y" fewer mistakes, then the information has saved "XY" dollars, so that would be its value. But I don't know what "X" and "Y" are, and I don't know how to find out.

TEST MANAGER

GOAL: My goal is to ensure that my testers find most of the defects and that very few escape into production.

QUESTIONS:

- What percentage of defects are being found during system testing, before system testing, and after system testing—that is, in production?
- What test techniques are effective at finding defects?
- How did the defects that escaped get by us?
- Where are the holes in our testing net?

METRICS:

- Defect detection percentage
- Defects found by test technique
- Defect escape analysis

VALUE: If the percentage of defects we find during testing is very high compared to those that escape into production, that makes my test team look good. But I know how this will play out. My management won't see any real value in this information; in fact, they will probably use it against me. They will quickly forget how many defects we're finding and just focus on those we've missed: "That was a very critical defect. How did you miss that one? It's apparent your team can only find the unimportant bugs. Novice users find the real ones." Perhaps we'd all be better off not reporting this information.

Relevant

Lookout on the Titanic: "Look, Captain, the northern lights. Aren't they beautiful tonight?"

The world is full of interesting, accurate, timely, complete information that is not relevant to the decisions at hand—don't include it in the information you supply to your clients.

Unique

Lookout on the Titanic: "It's cold out here tonight, Captain."

Information that is already known or that is readily available from other sources is generally not valued. It's just redundant. However, receiving the same information from multiple sources may be an indicator of its importance.

Actionable

Lookout on the Titanic: "Captain! Icebergs on the starboard side, coming at us fast!"

Important decisions can be made based on this information. Our clients are choosing from a number of actions, and this information allows them to make more effective choices.

Client-oriented

Finally, the information must be expressed in terms that are meaningful and useful to our clients including loss of sales, revenue, market share, and reputation. This also means claims for compensation, restorative actions, repair, and redistribution are important. It is not useful to report the number of test cases written, test cases executed, Severity 2 defects found, or the myriad of internal process metrics we often dump on them.

Proving Our Worth—Quantifying Our Value

We may not perform the Parsaye "value of information" experiment, but we can still quantify the value of the information that we, as testers, provide. Let's listen in on the private thoughts of a developer, test manager, and project manager:

DEVELOPER

GOAL: My goal is to get my code out on time without making any real blunders in it; I don't want the customer finding big defects. I'll look foolish in front of the whole team, especially my manager.

QUESTIONS:

- How many defects have I created?
- Are my defects scattered randomly throughout the code or are they clumped in a few modules?
- Am I making random mistakes or am I creating the same type of defects again and again?

METRICS:

- Number of defects
- Defect distribution
- Defect root cause

VALUE: Quantifying the value of this information is difficult for me. Having information about the defects I create could help me make fewer mistakes in the future. If it costs "X" dollars to find and fix each one of my defects, and if I made "Y" fewer mistakes, then the information has saved "XY" dollars, so that would be its value. But I don't know what "X" and "Y" are, and I don't know how to find out.

TEST MANAGER

GOAL: My goal is to ensure that my testers find most of the defects and that very few escape into production.

QUESTIONS:

- What percentage of defects are being found during system testing, before system testing, and after system testing—that is, in production?
- What test techniques are effective at finding defects?
- How did the defects that escaped get by us?
- Where are the holes in our testing net?

METRICS:

- Defect detection percentage
- Defects found by test technique
- Defect escape analysis

VALUE: If the percentage of defects we find during testing is very high compared to those that escape into production, that makes my test team look good. But I know how this will play out. My management won't see any real value in this information; in fact, they will probably use it against me. They will quickly forget how many defects we're finding and just focus on those we've missed: "That was a very critical defect. How did you miss that one? It's apparent your team can only find the unimportant bugs. Novice users find the real ones." Perhaps we'd all be better off not reporting this information.

PROJECT MANAGER

GOAL: My goal is a high-quality product that gives our customers a good experience while solving their problems—and doesn't create major post-release support costs.

QUESTIONS:

- How many defects have we found so far?
- What would their impact on our organization have been if they had escaped into production?
- How many more serious defects remain undiscovered?
- What would their impact be?

METRICS:

- Defect count
- Remaining defect estimate
- Defect impact analysis

VALUE: What if we could add up the dollar impact of each of the defects, had they escaped to the user and caused problems in the field, then subtract the cost of finding them? That would be the value of the information that testing created for the organization (assuming others acted on that information to improve the product's quality).

Of course, computing the dollar impact of the defects would be difficult. It's not the impact on the customer that is important—it is the loss that we incur because we delivered the software with those defects. The loss has two dimensions. The first is the immediate costs we incur. That includes the support costs to process the defect, as well as the help desk and possibly on-site support. And don't forget about the costs of the testers who are researching the defect and the developers who are repairing the code, then the cost of retesting, and finally the cost of distributing the software to our customers again. The second dimension is the long-term loss of revenue caused by the defect. This is much more difficult to quantify, since the loss may not come until months or even years later—for example, when the customer dumps our product in favor of the competitors', chiding us about the "Big Problem of '04."

These three scenarios remind us that quantifying the value of testing is difficult work. Perhaps that's why we concentrate so much on the testing process—it's much easier. But until we do this difficult work of proving our worth through quantifying our contribution, we should expect the bombardments to continue. {end}

Lee Copeland has more than thirty years' experience in the field of software development and testing. He has worked as a programmer, development director, process improvement leader, and consultant. Based on his experience Lee has developed and taught a number of training courses focusing on software testing and development issues. He is a technical editor and regular columnist for Better Software *magazine and StickyMinds.com and is the author of* A Practioner's Guide to Software Test Design. *Contact Lee at lcopeland@sqe.com.*

Sticky Notes

For more on the following topics, go to www.StickyMinds.com/bettersoftware

- Acknowledgements
- References